Digital Signal Processing Fundamentals (5ESC0)

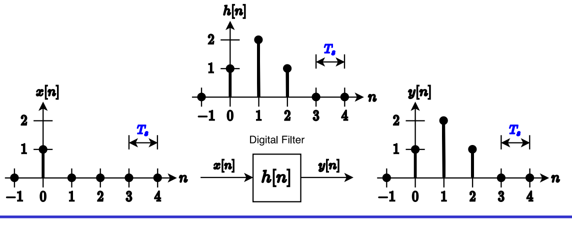
Discrete-time Signals and Systems

Sveta Zinger, Piet Sommen

(s.zinger@tue.nl)

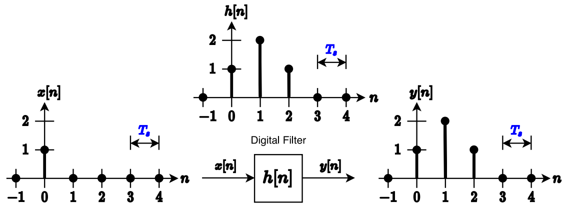
Discrete-time Signals and Systems

* We will be working in the digital domain, therefore we describe signal x as $x[n]$
* We use square brackets and n is the variable indicating a sample taken at n times the intersample distance



Discrete-time Signals and Systems

* The intersample distance depends on the time T_s between the taken samples, which depends on the sampling frequency f_s at which the analog to digital converter runs
* $x[n] \equiv x[n \cdot T_s]$ $h[n] \equiv h[n \cdot T_s]$ $y[n] \equiv y[n \cdot T_s]$
* Notation: we skip T_s if possible



Discrete-time signals

* Below are some fundamental sequences/signals that you have seen in Signals 1:
* The delta pulse, which has only a value of 1 for $n = 0$:

$$\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & \text{elsewhere} \end{cases}$$

* The unit step function (a.k.a. the Heaviside step function) has a value of 1 for all nonnegative n:

$$u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & \text{elsewhere} \end{cases}$$

Discrete-time signals

- The delta pulse and unit step function can be used to describe each other:

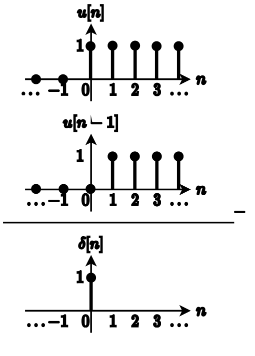
$$u[n] = \sum_{k=-\infty}^{n} \delta[k]$$

$$\delta[n] = u[n] - u[n-1]$$

Discrete-time signals

- Explanation for

$$\delta[n] = u[n] - u[n-1]$$



Discrete-time signals

* Explanation for

$$u[n] = \sum_{k=-\infty}^{n} \delta[k]$$

* For all $n < 0$, the values for k will be less than zero, hence the delta function will give a value 0 and the sum of these values $u[n]$ will be 0.
* For all $n \geq 0$, k will be 0 for exactly one of the summed delta functions as k runs from $-\infty$ to some positive n. The delta function $\delta[k]$ where $k = 0$ will give a value of 1 and for all $k > 0$, the delta function will be 0 again. The sum of all these values will be 1. Therefore $u[n] = 1$ for $n \geq 0$.

Discrete-time signals

* Below are some fundamental sequences/signals:
* An exponentially decaying function, which we will use later:

$$x[n] = a^n \cdot u[n] \text{ with } |a| < 1$$

* The discrete complex exponential function:

$$e^{jn\omega_0} = \cos(n\omega_o) + j\sin(n\omega_0)$$

Discrete-time signals

* We will also discuss Periodic and Aperiodic sequences/signals:
* In the digital domain a signal is periodic if for some integer N

$$x[n] = x[n+N]$$

Periodic and aperiodic signal examples

* Periodic if $x[n] = x[n+N]$ for some integer N
* Example of a periodic signal: $x_1[n] = \sin(\frac{n\pi}{4})$
* Take $N = 8$

$$x_1[n] = \sin(\frac{(n+8)\pi}{4}) = \sin(\frac{n\pi}{4} + 2\pi) = \sin(\frac{n\pi}{4})$$

* This signal has a period of 8 samples, so it will repeat its behavior every 8 samples

Periodic and aperiodic signal examples

* Periodic if $x[n] = x[n + N]$ for some integer N
* Example Aperiodic: $x_2[n] = \sin(n)$
* To be periodic $\sin(n + N)$ has to equal $\sin(n + 2\pi)$ for some N
* It follows that N has to equal 2π, which is impossible because π is irrational and N is an integer

* If we look at the function $f(x) = \sin(x)$ which we know from mathematics is periodic, but in the digital domain $x_2[n] = \sin(n)$ is not periodic because of how we sample

Discrete-time signals

* Signals or sequences can have forms of symmetry:

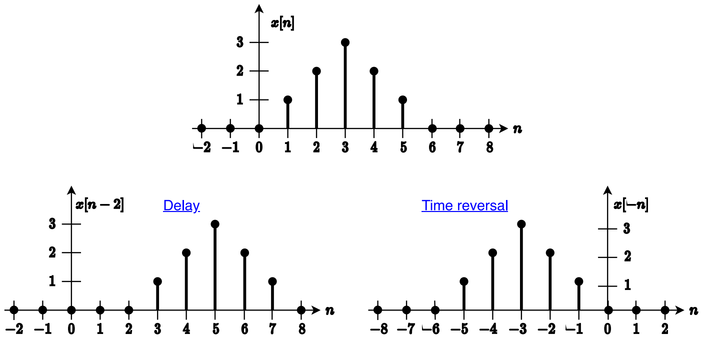
* Even : $x[n] = x[-n]$
  – Example: $x[n] = \cos[n]$
* Odd : $x[n] = -x[-n]$
  – Example: $x[n] = \sin[n]$
* Conjugate-symmetric : $x[n] = x^*[-n]$
* Conjugate-antisymmetric : $x[n] = -x^*[-n]$

Discrete-time signals

- Transformation of function variable: $y[n] = x[f[n]]$
- This means that we have an $x[n]$ and we change the variable n to find $y[n]$ by using some function $f[n]$
- Examples:
  - Time shifting (delay or advance) : $f[n] = n - n_0$
  - Time reversal : $f[n] = -n$
- As you may have already seen, time shifting is useful in Fourier transforms and time reversal is used in convolutions

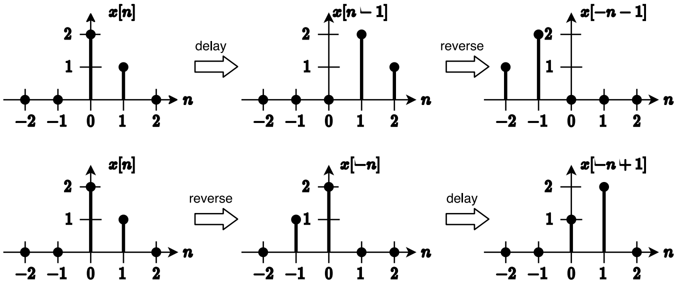
Discrete-time signals

- The figure below shows the signal $x[n]$ together with a delayed and time reversed version



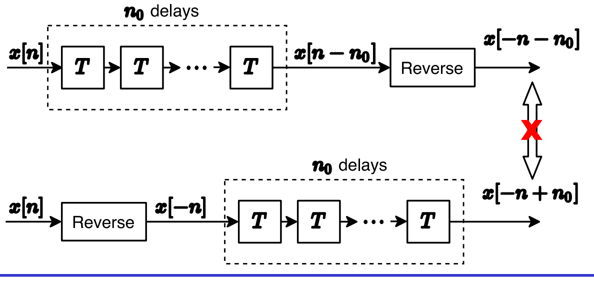
Discrete-time signals

* Shifting, Reversal and time-scaling are order dependent
* Because the outcomes of the same operations in a different order are not the same, there is order dependency
* This is illustrated in the figure below



Discrete-time signals

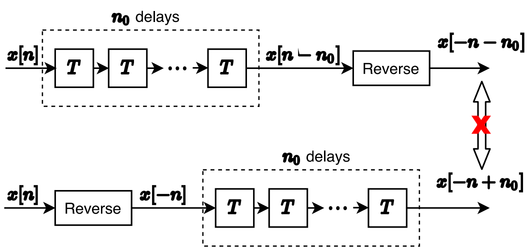
* Shifting, Reversal and time-scaling are order dependent
* We can see that the two block schemes below do not yield the same outcome by following the mathematics



Discrete-time signals

* $x[n] \xrightarrow{\text{delay}} x[n-n_0] \xrightarrow{\text{reverse}} x[(-n)-n_0] = x[-n-n_0]$
* $x[n] \xrightarrow{\text{reverse}} x[-n] \xrightarrow{\text{delay}} x[-(n-n_0)] = x[-n+n_0]$

* When a signal is reversed, only the running index is reversed



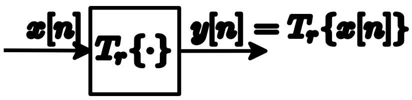
Discrete-time signals

* Operations we can use on discrete-time signals:

Addition	:	$y[n] = x_1[n] + x_2[n]$
Multiplication	:	$y[n] = x_1[n] \cdot x_2[n]$
Scaling	:	$y[n] = c \cdot x[n]$
Signal decomposition	:	$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k]$

* Signal decomposition: any digital signal can be written as a sum of weighted delta pulses

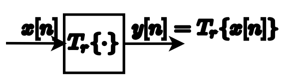
Discrete-time systems



* In general, we apply some transformation on a signal by using systems which can be characterized by the block scheme above

* It has the following properties

Discrete-time systems



* Properties:

Memoryless	:	Output at $n = n_0$ depends only on input at $n = n_0$
Additivity	:	$Tr\{x_1[n] + x_2[n]\} = Tr\{x_1[n]\} + Tr\{x_2[n]\}$ The result of adding two signals at the input is the same as the sum of their respective outputs
Homogeneity	:	$Tr\{c \cdot x[n]\} = c \cdot Tr\{x[n]\}$ A scalar multiplication at the input yields a scalar multiplication at the output
Linearity	:	$Tr\{a_1 x_1[n] + a_2 x_2[n]\} = a_1 Tr\{x_1[n]\} + a_2 Tr\{x_2[n]\}$ A combination of additivity and homogeneity
Time-Invariance	:	$y[n] = Tr\{x[n]\} \;\Rightarrow\; y[n - n_0] = Tr\{x[n - n_0]\}$ A shift at the input yields the same shift at the output

Discrete-time systems

$$x[n] \rightarrow \boxed{T_r\{\cdot\}} \rightarrow y[n] = T_r\{x[n]\}$$

* <u>Properties:</u>

LTI	:	Linear Time-Invariance (book: “Shift”-Invariance) Additive, Homogeneous and Time-Invariant
Causality	:	Response at n_0 depends on input up to $n = n_0$ In practice we cannot predict sample values, therefore it is important to design causal filters
(BIBO) Stability	:	For $A, B < \infty$, $\lvert x[n]\rvert < A \;\; \Rightarrow \;\; \lvert y[n]\rvert < B$ Input bounded by some number A will yield output bounded by some number B
Invertibility	:	Input may be uniquely determined from output Only one input can be traced back from the output

Important LTI properties

* Impulse response : $h[n] = Tr\{\delta[n]\}$
  This is a system’s response to a delta pulse input

* BIBO Stability : $\sum_{k=-\infty}^{\infty} \lvert h[k]\rvert < \infty$
  LTI systems are BIBO Stable if the sum of the impulse response is finite

* Convolution sum :

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] = x[n] * h[n]$$

Example LTI systems

Determine whether the following system is LTI

$$y[n] = au[n] + b$$

with constant parameters a and b

Solution:

We will check whether the system is linear and time-invariant

- Linearity check

The system must have the linearity property:

$$c_1u_1[n] + c_2u_2[n] \rightarrow \boxed{\mathrm{Tr}\{.\}} \rightarrow c_1y_1[n] + c_2y_2[n]$$

where c_1 and c_2 are real coefficients

Example LTI systems

For the input signal equal to $c_1u_1[n] + c_2u_2[n]$, the output will be

$$y[n] = au[n] + b = a(c_1u_1[n] + c_2u_2[n]) + \mathrm{b} = ac_1u_1[n] + ac_2u_2[n] + \mathrm{b}$$

If the system is linear, we should obtain the same output signal $(ac_1u_1[n] + ac_2u_2[n] + \mathrm{b})$ when considering the sum of outputs $c_1y_1[n] + c_2y_2[n]$, where

$$y_1[n] = au_1[n] + b$$
$$y_2[n] = au_2[n] + b$$

Example LTI systems

Let's now calculate the sum of these outputs:

$$y[n] = c_1 y_1[n] + c_2 y_2[n] = c_1 a u_1[n] + c_1 b + c_2 a u_2[n] + c_2 b$$
$$= a(c_1 u_1[n] + c_2 u_2[n]) + (c_1 + c_2) b$$

The expression above is not equal to the previously obtained result

$(a c_1 u_1[n] + a c_2 u_2[n] + \mathrm{b})$, so the system is not linear.

Example: system stability

Determine which of the following systems are stable:

a) $y[n] = x^2[n]$

b) $y[n] = cos(x[n])$

c) $y[n] = x[n] * cos(n\pi/8)$

<u>Solution</u>

When is a system stable?

BIBO stability means Bounded-Input, Bounded-Output.

Example: system stability

If a system is BIBO stable, then the output is bounded for every input to the system that is bounded:

$$|x[n]| < \infty \;\rightarrow |y[n]| < \infty$$

System (a)

Let's consider the first proposed system: $y[n] = x^2[n]$.

Let $x[n]$ be any bounded input with $|x[n]| < M$.

Then the output, $y[n] = x^2[n]$, will be bounded as well: $|y[n]| = |x[n]|^2 < M^2$.

Therefore, the system is stable.

Example: system stability

System (b)

The second proposed system is $y[n] = cos(x[n])$.

As $|\cos(x)| \leq 1$ for all x, this system is stable.

System (c)

Let's consider the third system: $y[n] = x[n] * cos(n\pi/8)$.

We can see that in this system we are dealing with convolution (marked as "$*$").

It is an LTI system, because in this system the output is defined as a convolution of input with impulse response:

$$\underbrace{y[n]}_{\text{output signal}} = \underbrace{x[n]}_{\text{input signal}} * \underbrace{cos(n\pi/8)}_{\text{impulse response}}$$

Example: system stability

For an LTI system to be stable, the following condition should be held for its impulse response:

$$\sum_{k=-\infty}^{\infty} |h(k)| < \infty$$

In our case, $cos(n\pi/8) = h[k]$. We can observe, that $cos(n\pi/8)$ can be depicted as 

Sampling this graph and letting k vary from $-\infty$ to ∞ will lead to infinitely large numbers.

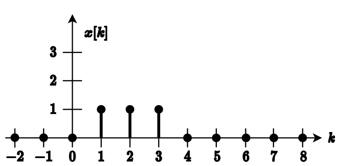
So we can say that this system is unstable.

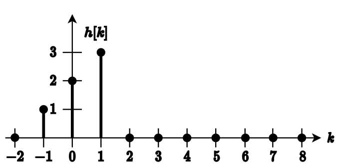
Convolution procedure

* Convolution sum:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

* For finite length sequences: $\text{Length}(y) = \text{Length}(x) + \text{Length}(h) - 1$
* We have input signal $x[n]$ and impulse response $h[n]$
* For the convolution sum we switch n to k

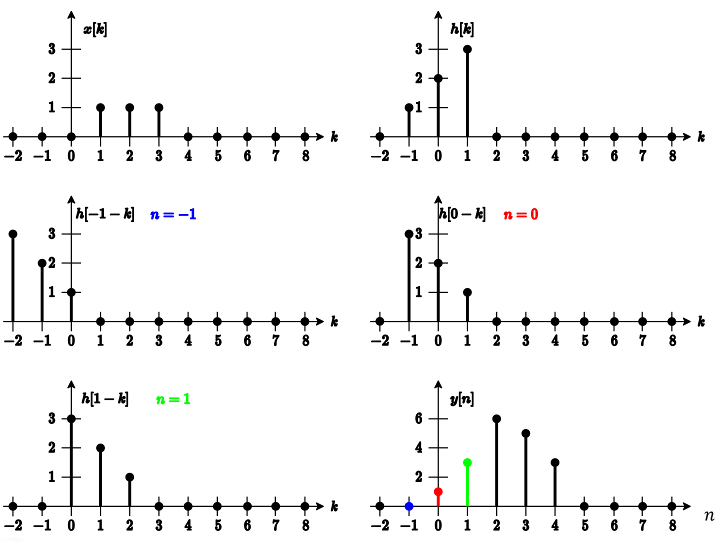




Convolution procedure

* We then reverse $h[k]$ and shift by n to obtain $h[n-k]$, where n are the samples of $y[n]$
* $h[k] \xRightarrow{\text{reverse}} h[-k] \xRightarrow{\text{delay}} h[-(k-n)] = h[-k+n] = h[n-k]$
* We then multiply the coefficients of the overlapping samples of $h[n-k]$ and $x[k]$ and sum these to find $y[n]$

Convolution procedure



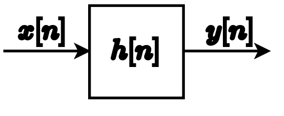
Convolution procedure

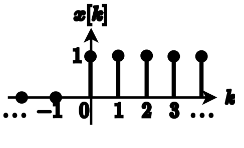
n	-2	-1	0	1	2	3	4	5	6	$\sum_{k=-\infty}^{\infty} x[k]h[n-k]$
$x[k]$	0	0	0	1	1	1	0	0	0	
$h[-1-k]$	3	2	1	0	0	0	0	0	0	0
$h[0-k]$	0	3	2	1	0	0	0	0	0	1
$h[1-k]$	0	0	3	2	1	0	0	0	0	3
$h[2-k]$	0	0	0	3	2	1	0	0	0	6
$h[3-k]$	0	0	0	0	3	2	1	0	0	5
$h[4-k]$	0	0	0	0	0	3	2	1	0	3
$h[5-k]$	0	0	0	0	0	0	3	2	1	0

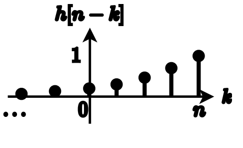
* $y[n] = \delta[n] + 3\delta[n-1] + 6\delta[n-2] + 5\delta[n-3] + 3\delta[n-4]$

Convolution example

* $x[n] = u[n]$
* $h[n] = a^n\, u[n], \quad |a| < 1$



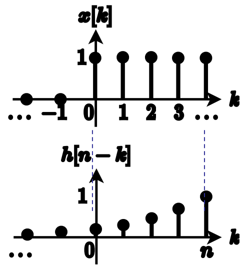




* Find $y[n]$

Convolution example

* $x[n] = u[n]$ $\quad\quad\quad\quad h[n] = a^n\, u[n], \quad |a| < 1$
* $h[n-k] = a^{n-k} u[n-k]$
* Graphical solution: look at the overlap
* There is only overlap for $k = 0$ to $k = n$
* For each of the overlapping values $x[k] = 1$ and $h[n-k] = a^{n-k}$
* $y[n] = \sum_{k=0}^{n} a^{n-k}$
* $= a^n \sum_{k=0}^{n} (a^{-1})^k$ (series that we know)
* $y[n] = a^n \cdot \frac{1-a^{-(n+1)}}{1-a^{-1}} = \frac{1-a^{n+1}}{1-a}$



Convolution example

* $x[n] = u[n]$ $\quad\quad\quad\quad h[n] = a^n\, u[n], \quad |a| < 1$
* $h[n-k] = a^{n-k} u[n-k]$
* Mathematical solution: fill in the convolution sum
* $y[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k]$
* $\sum_{k=-\infty}^{\infty} x[k] h[n-k] = \sum_{k=-\infty}^{\infty} u[k] a^{n-k} u[n-k]$
* We change the bounds to where the unit step functions are 1
* $\sum_{k=-\infty}^{\infty} u[k] a^{n-k} u[n-k] = \sum_{k=0}^{\infty} a^{n-k} u[n-k] = \sum_{k=0}^{n} a^{n-k}$
* $y[n] = \frac{1-a^{n+1}}{1-a}$

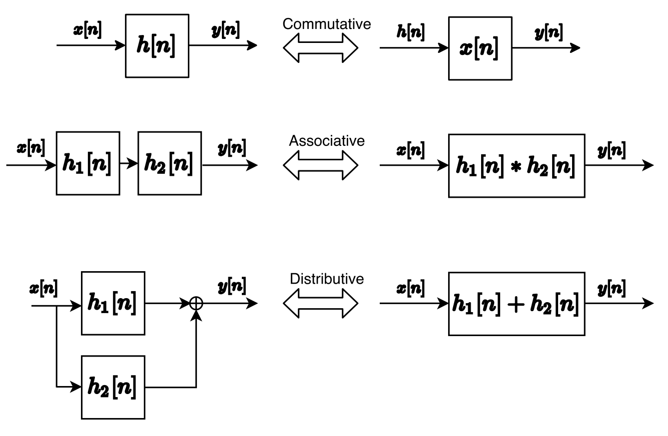
Convolution properties

* Commutative: $x[n] * h[n] = h[n] * x[n]$
  If we visualize the overlap of $x[n]$ and $h[n]$, then the overlap of $x[n]$ over $h[n]$ is the same as the overlap of $h[n]$ over $x[n]$. This is useful when computing convolutions, because you can choose which signal you reverse.

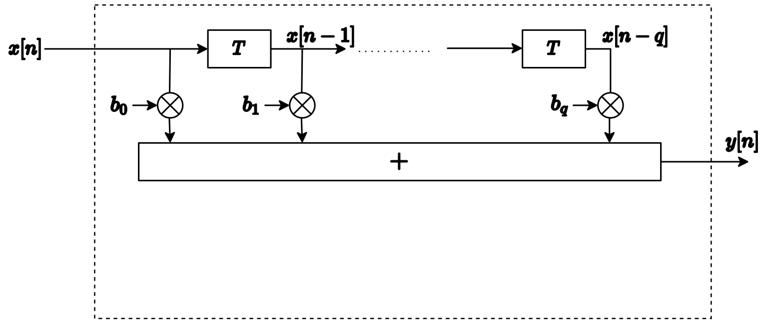
* Associative: $\{x[n] * h_1[n]\} * h_2[n] = x[n] * \{h_1[n] * h_2[n]\}$
  The overall impulse response of cascaded filters is the convolution of their respective impulse responses.

* Distributive: $x[n] * h_1[n] + x[n] * h_2[n] = x[n] * \{h_1[n] + h_2[n]\}$
  The overall impulse response of parallel filters is the sum of their respective impulse responses.

Convolution properties



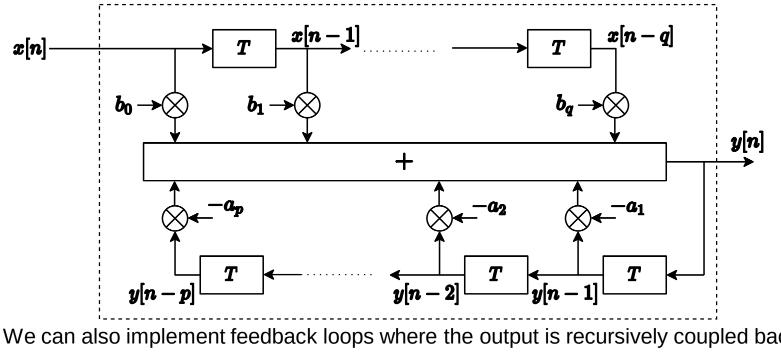
Signal flow diagram LTI system



- Also known as a realization scheme
- Any Finite Impulse Response (FIR) filter can be represented in this way
- Difference Equation (DE):

$$y[n] = \sum_{k=0}^{q} b_k x[n-k]$$

Signal flow diagram LTI system



- We can also implement feedback loops where the output is recursively coupled back
- Infinite Impulse Response (IIR)
- Difference Equation (DE):

$$y[n] = \sum_{k=0}^{q} b_k x[n-k] - \sum_{k=1}^{p} a_k y[n-k]$$

Difference equations (DE)

$$y[n] = \sum_{k=0}^{q} b_k x[n-k] - \sum_{k=1}^{p} a_k y[n-k]$$

* FIR=Finite Impulse Response: All $a_k = 0$ → *Nonrecursive DE*

$$y[n] = \sum_{k=0}^{q} b_k x[n-k] \;\Rightarrow\; h[n] = \sum_{k=0}^{q} b_k \delta[n-k]$$

* IIR=Infinite Impulse Response: At least one $a_k \neq 0$ → *Recursive DE*

* Different methods to solve DE:
  1. Evaluate DE for each different n
  2. Classical approach via homogeneous and particular solutions
  3. Use Z-transform (chapter 4)

Summary

* We considered discrete-time signals:
  – Denoted by square brackets
  – Some fundamental signals: delta pulse, unit step function, exponentially decaying function and complex exponential
  – Periodicity, symmetry, transformation of function variable and order dependency

* And discrete-time systems:
  – Properties
  – LTI properties
  – Convolution
  – Convolution properties
  – Signal flow diagrams and difference equations